500.0250
8605

PATENT

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Applicant(s): Frazier (et al.)

Serial No.: 09/782,151

Filed: February 14, 2001

For: OPERATING SOFTWARE SCHEDULING PRIORITY READER

Group: 2122

Examiner: Yigdall, Michael J.

Durham, North Carolina
October 30, 2006

Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

CERTIFICATION OF FACSIMILE TRANSMISSION

Sirs:

I hereby certify that this correspondence is being facsimile transmitted to the United States Patent and Trademark Office, Fax. No. 571-273-8300 on the date set forth below

1. Transmittal of Appellants' Brief (1 page);
2. Appellants' Brief (24 pages);
3. Amendment After Final (8 pages)

Marianna Tortorelli
Printed name of person signing

Marianna Tortorelli
Signature

Date: October 30, 2006

RECEIVED
CENTRAL FAX CENTER
OCT 30 2006

500.0250
8605

PATENT

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Applicants: Frazier et al.

Serial No.: 09/782,151

Filed: February 14, 2001

For: OPERATING SOFTWARE SCHEDULING PRIORITY RECORDER

Group: 2122

Examiner: Yigdall, Michael J.

Durham, North Carolina
October 30, 2006

MAIL STOP APPEAL BRIEF – PATENTS
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

TRANSMITTAL OF APPELLANTS' BRIEF

Dear Sirs:

1. Transmitted herewith is the APPEAL BRIEF in this application with respect to the Notice of Appeal filed on August 31, 2006.

2. The Applicant is other than a small entity.

3. Pursuant to 37 CFR 1.17(f) the fee for filing the Appeal Brief is $500.00.

[x] The Commissioner is hereby authorized to charge the fee of $500 to NCR Deposit Account No. 14-0225.

[x] The Commissioner is hereby authorized to charge any additional fees which may be required or credit any overpayment to NCR Deposit Account No. 14-0225.

Respectfully submitted,

10/31/2006 TL0111 00000055 140225 09782151
01 FC:1402 500.00 DA

Peter H. Priest
Reg. No. 30,210
Priest & Goldstein, PLLC
5015 Southpark Drive, Suite 230
Durham, NC 27713
(919) 806-1600

RECEIVED
CENTRAL FAX CENTER
OCT 3 0 2006

500.0250
8605

PATENT

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE
BEFORE THE BOARD OF PATENT APPEALS AND INTERFERENCES

In re Application of	:	Frazier et al.
For	:	Operating Software Scheduling Priority Reader
Serial No.	:	09/782,151
Filed	:	02/14/2001
Group	:	2192
Examiner	:	Yigdall, Michael J.

Durham, North Carolina
October 30, 2006

MAIL STOP APPEAL BRIEF – PATENTS
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

APPELLANTS' BRIEF

Sir:

1. The Real Party In Interest

The real party in interest is the assignee, NCR Corporation.

2. Related Appeals and Interferences

None.

RECEIVED
CENTRAL FAX CENTER
OCT 3 0 2006

3. Status of the Claims

This is an appeal from the May 31, 2006 final rejection ("the final rejection") of claims 1-20, all of the pending claims. Claims 1 and 3-17 were rejected under 35 U.S.C. § 103(a) as being unpatentable over Bishop U.S. Patent No. 6,049,798 ("Bishop") in view of Barritz U.S. Patent No. 5,570,056 ("Barritz") in view of Farrell U.S. Patent No. 5,247,675 ("Farrell") in view of Yamagishi U.S. Patent No. 5,870,604 ("Yamagishi"). Claims 2 and 18-20 were rejected under 35 U.S.C. § 103(a) as being unpatentable over Bishop in view of Barritz in view of Farrell and in view of Yamagishi, and further in view of O'Brien U.S. Patent No. 6,795,809 ("O'Brien"). Pending claims 1-20 are the subject of this appeal.

4. Status of Amendments

An amendment is being filed herewith in order to correct a lack of antecedent basis in claims 19 and 20. Otherwise, the claims stand as last amended on March 8, 2006.

5. Summary of Claimed Subject Matter

The present invention addresses systems and techniques for recording software program scheduling information from within an operating system as addressed further below.

Claim 1

Claim 1 addresses a method of capturing operating software scheduling information during execution of operating software and recording said operating software scheduling information from within the operating software. The method comprises the steps of compiling operating software scheduling information capture software which is an integral part of the operating system. See specification, p. 9, lines 15-20, for example. The operating software scheduling information capture software records a history of operating software events as they occur. See specification, p. 10, lines 26-28, for example. Information related to the history is

organized and stored as operating software program scheduling information relating to interactions between the operating system software and each of the programs and tasks managed by the operating system software. See specification, p. 10, lines 20-25, for example. The scheduling information includes indications of relative priorities of programs and tasks, indications of transfers of control from lower priority to higher priority tasks and identification of tasks waiting for execution at the occurrence of each operating software event. See specification, p. 13, line 5-p. 14, line 20, for example. The method further comprises invoking operating software scheduling information capture. See specification, p. 13, lines 15-23, for example. The method further comprises recording operating software scheduling information while the operating software is executing, for relatively long duration storage in order to permit review of the scheduling information by a user, duration of storage being sufficient to allow data collected during an operating session to be retrieved and used after termination of the session. See specification, p. 13, lines 15-28, for example.

Claim 15

Claim 15 addresses a method of capturing operating software scheduling information during execution of said operating software and recording said operating software scheduling information from within the operating software, wherein said method is performed using operating software scheduling information compiled and integrated with the operating software. . See specification, p. 9, lines 15-20, for example. The method comprises invoking operating software scheduling information capture software. The operating software scheduling information capture software records a history of operating software events as they occur. See specification, p. 10, lines 26-28, for example. The history is organized and stored as operating software program scheduling information relating to interactions between the operating system

software and each of the programs and tasks managed by the operating system software. See specification, p. 13, line 5-p. 14, line 20, for example. The scheduling information includes indications of relative priorities of programs and tasks, indications of transfers of control from lower priority to higher priority tasks and identification of tasks waiting for execution at the occurrence of each operating software event. See specification, p. 13, line 5-p. 14, line 20, for example. The method further comprises recording operating software scheduling information while the operating software is executing for relatively long duration storage in order to permit review of the scheduling information by a user, duration of storage being sufficient to allow data collected during an operating session to be retrieved and used after termination of the session. See specification, p. 13, line 5-p. 14, line 20, for example.

Claim 18

Claim 18 addresses a retail store bar code scanner employing operating software and operating software scheduling information capture software which operates during execution of said operating software to record a history of operating software events as they occur. The scanner comprises means for scanning a specific bar code tag sequence. See, specification, p. 12, lines 6-11, p. 10, lines 26-28, for example. The scanner further comprises a processor for receiving and transmitting data and a memory coupled to the processor. See specification, p. 6, lines 14-25, for example. The memory has stored therein sequences of instructions which, when executed by the processor, cause the processor to invoke operating software scheduling information capture software. See specification, p. 13, lines 15-23, for example. The operating software scheduling information capture software is operative to record a history of operating software events as they occur. See specification, p. 10, lines 26-28, for example. The history is organized and stored as operating software program scheduling information relating to

interactions between the operating system software and each of the programs and tasks managed by the operating system software. See specification, p. 13, line 5-p. 14, line 20, for example. The scheduling information includes indications of relative priorities of programs and tasks, indications of transfers of control from lower priority to higher priority tasks and identification of tasks waiting for execution at the occurrence of each operating software event. See specification, p. 13, line 5-p. 14, line 20, for example. The operating software scheduling information capture software is further operative to record operating software scheduling information for relatively long duration storage in order to permit review of the scheduling information by a user. Duration of storage is sufficient to allow data collected during an operating session to be retrieved and used after termination of the session wherein the specific bar code tag sequence selects the information to be recorded by the operating software scheduling information capture software. See specification, p. 12, lines 6-11, p. 13, line 5-p. 14, line 20, for example.

6. Grounds of Rejection to be Reviewed on Appeal

Claims 1 and 3-17 stand rejected under 35 U.S.C. § 103(a) as being unpatentable over Bishop, Barritz, Farrel, and Yamagishi. Claims 2 and 18-20 stand rejected under 35 U.S.C. § 103(a) in view of Bishop, Barritz, Farrel, and Yamagishi in view of O'Brien.

7. Argument

The final rejection under 35 U.S.C. § 103 did not follow M.P.E.P. § 706.02(j) which states:

> After indicating that the rejection is under 35 U.S.C. 103, the Examiner should set forth...the difference or differences in the claim over the applied reference,...the proposed modification of the applied reference(s) necessary to arrive at the claimed subject matter, and ... an explanation why one of ordinary skill in the art at the time the invention was made would have been motivated to make the proposed modification.

As will be illustrated below, the claims of the present invention are not taught by and are not obvious in view of the references relied upon by the Examiner.

A. Rejection under 35 U.S.C. § 103 over Bishop, Barritz, Farrel, and Yamagishi

35 U.S.C. § 103 which governs obviousness indicates that "differences between the subject matter sought to be patented and the prior art" are to be assessed based upon "the subject matter as a whole". Analyzing the entirety of each claim, the rejections under 35 U.S.C. § 103 are not supported by the relied upon art as addressed further below.

Only after an analysis of the individual references has been made can it then be considered whether it is fair to combine teachings. However, as addressed further below, fairness requires an analysis of failure of others, the lack of recognition of the problem, and must avoid the improper hindsight reconstruction of the present invention. Such an analysis should consider whether the modifications are actually suggested by the references rather than assuming they are obvious.

The 35 U.S.C. § 103 rejections made here pick and choose elements from separate references, none of which presents any motivation for making the suggested combination. This approach constitutes impermissible hindsight and must be avoided. As required by 35 U.S.C. § 103, claims must be considered as a whole. When so considered, the present claims are not obvious.

Claim 1

Turning to the references relied upon, Bishop, Barritz, Farrel, and Yamagishi are markedly different from the present invention and address problems only peripherally related to the solutions provided by the present invention. Bishop teaches systems and techniques for monitoring resource usage in a system, Barritz teaches systems and techniques for collecting

usage frequency information for various software modules used by a computer system, in order to identify underused or obsolete software modules. Farrell teaches scheduling management in a multitasking system, assigning priorities to various program threads and using the priority information for execution scheduling. Yamagishi teaches job allocation in a multiprocessor system, identifying the number of jobs awaiting execution in order to determine when a job is to be transferred from one processor to another. Nothing in Bishop, Barritz, Farrell, or Yamagishi, however, teaches or makes obvious the relatively long term storage of operating software scheduling information including relative priorities of programs and tasks.

Claim 1 reads as follows:

1. A method of capturing operating software scheduling information during execution of operating software and recording said operating software scheduling information from within the operating software, the method comprising the steps of:

compiling operating software scheduling information capture software which is an integral part of the operating system, the operating software scheduling information capture software recording a history of operating software events as they occur, information related to the history being organized and stored as operating software program scheduling information relating to interactions between the operating system software and each of the programs and tasks managed by the operating system software, the scheduling information including indications of relative priorities of programs and tasks, indications of transfers of control from lower priority to higher priority tasks and identification of tasks waiting for execution at the occurrence of each operating software event;

invoking operating software scheduling information capture; and

recording operating software scheduling information while the operating software is executing, for relatively long duration storage in order to permit review of the scheduling information by a user, duration of storage being sufficient to allow data collected during an operating session to be retrieved and used after termination of the session.

These limitations in the claimed combination are not taught and are not made obvious by Bishop, Barritz, Farrell, Yamagishi, or a combination thereof.

Bishop teaches systems and techniques for monitoring of resources in a data processing system. Bishop monitors the overall level of resource usage, for example CPU and RAM usage, without concentrating on which individual tasks are using which resource. Bishop monitors

peripheral device usage by receiving information from device drivers. The device drivers sometimes provide information about the process using the device, but the information provided by device drivers, and the other resource usage information provided by Bishop, does not include a history of operating software events with information relating to the history being organized and stored as operating software program scheduling information relating to interactions between the operating system software and each of the programs and tasks managed by the operating system software, as is claimed by claim 1, as amended.

Bishop is directed toward the display of resource utilization in real time and contemplates the monitoring and displaying of resource utilization information during an operating session, with no indication that the collected information is to be preserved for retrieval or use beyond the session during which it is collected. Bishop's designation of a viewing period allows for a user of the system of Bishop to specify a time window for which information is to be presented, but any past information collected for presentation within the time window is simply part of a continuous information stream, preserved so as to provide context for a user. This context allows a user to identify trends and changes and to see the effects changing conditions, including changes to the operation of the system caused by user actions and adjustments. Bishop does not indicate that the viewing period remains open past the termination of an operating session, and does not indicate that the collected information is retrievable past the termination of an operating session.

The storage of scheduling information claimed by claim 1 allows for retrieval and examination of information over a number of sessions. Such retrieval and examination allows for assembly and analysis of statistical information indicating system performance over a time period involving a number of sessions, and for comparison of system events and performance

between different sessions. Analysis and comparisons made possible by preservation of scheduling information beyond the termination of a session allows for system refinements and improvements extending beyond those that can be performed during a single session. See, for example, the specification at p. 13, lines 17-23, which describes the retrieval of preserved information and notes the usefulness of such information in the development of robust embedded software, such as in the embodiment of a retail bar code scanner as addressed at page 12, lines 8-23, for example, and claimed in claims 2 and 18.

Adding Barritz to Bishop does not cure Bishop's deficiencies as a reference with respect to claim 1, as amended. Claim 1 claims that a history of operating system events organized and stored as scheduling information includes indications of relative priorities of programs and tasks, indications of transfers of control from lower priority to higher priority tasks and identification of tasks waiting for execution at the occurrence of each operating software event. Neither Bishop, Barritz nor a combination of Bishop and Barritz teaches or makes obvious this combination of features cooperating as claimed.

Bishop does not employ indications of relative priorities of programs and tasks. In order to track processor capacity that is not used by normally operating processes, Bishop begins a new process that is given a lower priority than any executing process on the system. Bishop notes the time employed by this lowest priority process in order to determine time that is not used by any of the normal processes on the system. The lowest priority process can only run when none of the normal processes are running, so the run time of the lowest priority process can be tracked and used to determine the normal idle time for the system, that is, the time when none of the normal processes on the system need to run. Bishop uses this lowest priority process to determine the overall system usage, and does not track the relative priorities of different

processes.

Adding Barritz to Bishop does not achieve and does not make obvious the invention as claimed by claim 1. Barritz teaches the collection of frequency of usage information for various software modules that may be hosted on a computer system. Barritz identifies a module when that module is invoked, identifying the particular software version represented by the module and matching module usage with licensing information in order to detect usage of unlicensed software. Barritz helps to identify underused and obsolete software modules in order to allow decisions to remove the underused and obsolete modules in order to save storage resources. Barritz operates to identify unlicensed usage in order to assist in complying with licensing requirements. Barritz does not address resource usage within a system or the allocation of system resources to different modules. Barritz simply addresses the number of times a particular module was used, in order to allow review. Bishop and Barritz do not address the same concerns, and the addition of Barritz to Bishop would not contribute to the information provided by Bishop. Bishop is concerned with resource usage, not with how often a particular software module is used. Logging the frequency of usage of software modules would not provide additional insight into the immediate loads on resources, which is the concern addressed by Bishop.

In addition, the information collected by Barritz does not include information relating to relative priorities of programs and tasks, transfers of control from lower priority to higher priority tasks and tasks waiting for execution at the occurrence of each operating software event. This information, collected by the present invention as claimed by claim 1 provides insight into the ongoing operation of a system and the allocation and management of resources to accomplish the various tasks performed by the system.

Adding Farrell to Bishop and Barritz does not teach and does not make obvious the invention as claimed by claim 1. Farrell teaches construction of threads having dispatch classes, and the storage and use of priority information in order to manage the assignment of priorities to threads and the allocation of processing resources among threads, but Farrell does not teach the relatively long term storage of scheduling information to permit review by a user, with the duration of storage being sufficient to allow data collected during an operating session to be retrieved and used after termination of the session. Storage of scheduling information to permit review by a user allows for insight by a user into the demands on and performance of a system and allows evaluation of the system to determine what adjustments, if any, need to be made, with the evaluation being able to take into account the demands and performance prevailing in multiple operating sessions and comparisons between demands and performance across multiple sessions. Such insights, and the relatively long term storage of information that permits the sort of review leading to such insights, are not provided by or relevant to Farrell, which simply uses priority information in operation but does not store it in order that it may be delivered for review by a user.

Adding Yamagishi to Bishop, Barritz and Farrell does not cure their deficiencies as references with respect to claim 1. Yamagishi distributes workloads among processors and uses data relating to the tasks awaiting execution and the number of tasks awaiting execution in order to manage workloads. Yamagishi does not collect data identifying the tasks awaiting execution at each particular software event for relatively long duration storage in order to permit review by a user, but simply uses the information in operation. Claim 1 claims recording scheduling information that includes identification of tasks awaiting execution at each software event so that the scheduling information may be presented to a user for review. Such collection and

presentation of information allows for insight into the operation of a system and helps to determine what adjustments, if any, need to be made. The relatively long term storage for presentation to a user is not accomplished by or relevant to Yamagishi, which simply uses information relating to the tasks or numbers of task awaiting execution in workload balancing, but does not contemplate that the information will be reviewed in order to provide insight by a user into the operation of the system.

Combining the references in the manner suggested would not be obvious, and would not achieve the invention claimed by claim 1. Bishop, Farrell, and Yamagishi use short term operational information to achieve various goals during operation of an operating system, such as monitoring resource usage, scheduling management, and allocating of jobs between processors. Bishop, Farrell, and Yamagishi collect and use information as it is needed to perform their goals, and do not preserve it beyond an operating session. Barritz preserves information beyond an operating session, but the information collected by Barritz does not relate to resource usage or priority between tasks. Barritz simply monitors how frequently various software modules are used, without attention to the loads they are placing on a system or priorities given to various modules. Barritz collects and preserves a different category of information from that collected by Bishop, Farrell, and Yamagishi, and adding the collection and preservation of Barritz's information to the collection and use of information by Bishop, Farrell, and Yamagishi would not contribute anything to the goals toward which Bishop, Farrell, and Yamagishi are directed. Claim 1 therefore defines over the cited art and should be allowed.

Claim 15

Claim 15 reads as follows:

15. A method of capturing operating software scheduling information during execution of said operating software and recording said operating software scheduling

information from within the operating software, wherein said method is performed using operating software scheduling information compiled and integrated with the operating software, the method comprising the steps of:

invoking operating software scheduling information capture software, the operating software scheduling information capture software recording a history of operating software events as they occur, the history being organized and stored as operating software program scheduling information relating to interactions between the operating system software and each of the programs and tasks managed by the operating system software, the scheduling information including indications of relative priorities of programs and tasks, indications of transfers of control from lower priority to higher priority tasks and identification of tasks waiting for execution at the occurrence of each operating software event; and

recording operating software scheduling information while the operating software is executing for relatively long duration storage in order to permit review of the scheduling information by a user, duration of storage being sufficient to allow data collected during an operating session to be retrieved and used after termination of the session.

For the reasons stated above with respect to claim 1, neither Bishop, Barritz, Farrell, Yamagishi, or a combination thereof teaches or makes obvious the relatively long duration storage of operating software scheduling information wherein the scheduling information includes indications of relative priorities of programs and tasks managed by operating system software. Claim 15 therefore defines over the cited art and should be allowed.

Claims 3-17

Claims 3-14 depend directly or indirectly from claim 1, incorporating all of the limitations thereof and adding further limitations thereto. Claims 16 and 17 depend directly from claim 15, incorporating all of the limitations thereof and adding further limitations thereto. In addition, the dependent claims address a number of combinations of limitations not found in the applied references. Claims 3-17 therefore define over the cited art and should be allowed.

B. Rejection under 35 U.S.C. § 103 over Bishop, Barritz, Farrell, and Yamagishi, and further in view of O'Brien

Claim 2

Claim 2 depends directly from claim 1, incorporating all of the limitations thereof and adding additional limitations thereto. O'Brien does not cure the deficiencies of Bishop, Barritz, Farrell, and Yamagishi as references.

Claim 18

Claim 18 reads as follows:

18. A retail store bar code scanner employing operating software and operating software scheduling information capture software which operates during execution of said operating software to record a history of operating software events as they occur, the scanner comprising:

means for scanning a specific bar code tag sequence;

a processor for receiving and transmitting data; and

a memory coupled to the processor, the memory having stored therein sequences of instructions which, when executed by the processor, cause the processor to invoke operating software scheduling information capture software, the operating software scheduling information capture software being operative to record a history of operating software events as they occur, the history being organized and stored as operating software program scheduling information relating to interactions between the operating system software and each of the programs and tasks managed by the operating system software, the scheduling information including indications of relative priorities of programs and tasks, indications of transfers of control from lower priority to higher priority tasks and identification of tasks waiting for execution at the occurrence of each operating software event, and to record operating software scheduling information for relatively long duration storage in order to permit review of the scheduling information by a user, duration of storage being sufficient to allow data collected during an operating session to be retrieved and used after termination of the session wherein the specific bar code tag sequence selects the information to be recorded by the operating software scheduling information capture software.

As noted above with respect to claim 1, neither Bishop, Barritz, Farrell, Yamagishi, or a combination thereof teaches or makes obvious the relatively long duration storage of operating software scheduling information wherein the scheduling information includes indications of relative priorities of programs and tasks managed by operating system software. O'Brien teaches reading of bar code information and taking actions, such as logging information, in response to

reading specific sequences of bar codes, but does not address the collection and storage of operating software scheduling information as claimed by claim 18. O'Brien therefore does not cure the deficiencies of Bishop, Barritz, Farrell, and Yamagishi as references, and claim 18 therefore defines over the cited art and should be allowed.

Claims 19 and 20

Claims 19 and 20 depend directly from claim 18, incorporating all of the limitations thereof and adding further limitations thereto. In addition, the dependent claims address a number of combinations of limitations not found in the applied references. Claims 19 and 20 therefore define over the cited art and should be allowed.

C. The Examiner's Findings of Obviousness Are Also Contrary to Law of the Federal Circuit

As shown above, the invention claimed is not suggested by the relied upon prior art. The references cited by the Examiner, if anything, teach away from the present invention. It is only in hindsight, after seeing the claimed invention, that the Examiner could combine the references as the Examiner has done. This approach is improper under the law of the Federal Circuit, which has stated that "[w]hen prior art references require selective combination by the Court to render obvious a subsequent invention, there must be some reason for the combination other than the hindsight gleaned from the invention itself." Uniroyal, Inc. v. Rudkin-Wiley Corp., 837 F.2d 1044, 1051, 5 U.S.P.Q. 2d 1434, 1438 (Fed. Cir. 1988), cert. den., 109 S. Ct. 75, 102 L.Ed. 2d 51 (1988); quoting Interconnect Planning Corp. v. Feil, 774 F.2d 1132, 1132, 227 U.S.P.Q. 543, 535 (Fed. Cir. 1985). Furthermore, "[i]t is impermissible to use the claims as a frame and the prior art references as a mosaic to piece together a facsimile of the claimed invention." Uniroyal, 837 F.2d at 1051, 5 U.S.P.Q. 2d at 1438. Similarly, "[t]he mere fact that the prior art could be so modified would not have made the modification obvious unless the prior art suggested the

desirability of the modification." In re Laskowski, 871 F.2d 115, 117, 10 U.S.P.Q. 2d 1397, 1398 (Fed. Cir. 1989), quoting In re Gordon, 733 F.2d 900, 902, 221 U.S.P.Q. 1125, 1127 (Fed. Cir. 1984). No such suggestion is found here.

In addition, the Examiner does not appear to have considered "where the references diverge and teach away from the claimed invention", Akzo N.V. v. International Trade Commission, 808 F.2d 1471, 1481, 1 U.S.P.Q. 2d 1241, 1246 (Fed. Cir. 1986), cert. den., 107 S. Ct. 2490, 482 U.S. 909, 107 S.Ct. 2490 (1987); and W.L. Gore Associates, Inc., 721 F.2d 1540, 220 U.S.P.Q. 303 (Fed. Cir. 1983); nor has the Examiner read the claims as a whole, as required by statute. 35 U.S.C. §103. See also, Smithkline Diagnostics Inc. v. Helena Laboratories Corp., 859 F.2d 878, 885, 8 U.S.P.Q. 2d 1468, 1475 (Fed. Cir. 1988); and Interconnect Planning Corp., 774 F.2d at 1143, 227 U.S.P.Q. at 551.

In In re Laskowski, 871 F.2d 115, 10 U.S.P.Q. 2d 1397, the Federal Circuit reversed an obviousness rejection of the claims in an application for a bandsaw. The claimed bandsaw used a pulley type wheel loosely fitted with a tire. The primary reference showed a similar bandsaw where the band was tightly fitted. The Federal Circuit stated that the prior art did not provide a suggestion, reason or motivation to make the modification of the reference proposed by the Commissioner. Id. at 1398. The Court added that "there must be some logical reason apparent from the positive, concrete evidence of record which justifies a combination of primary and secondary references." Id. quoting In re Regel, 526 F.2d 1399, 1403, 188 U.S.P.Q. 136, 139 (C.C.P.A. 1975), citing In re Stemniski, 444 F.2d 581, 170 U.S.P.Q. 343 (C.C.P.A. 1971).

In Uniroyal Inc. v. Rudkin-Wiley Corp., 837 F.2d 1044, 5 U.S.P.Q. 2d 1434 (Fed. Cir. 1988), cert. den., 109 S. Ct. 75, 102 L.Ed. 2d 51 (1988), the Federal Circuit reversed the District Court's finding that the claims for a patent for an air flow deflecting shield were obvious.

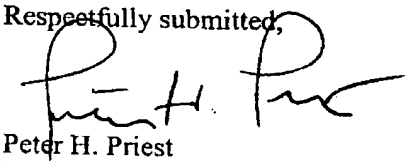
Without any suggestion in the art, the District Court improperly chose features from several prior art references to recreate the claimed invention.

The Examiner's rejection suggests that the Examiner did not consider and appreciate the claims as a whole. The claims disclose a unique combination with many features and advantages not shown in the art. It appears that the Examiner has oversimplified the claims and then searched the prior art for the constituent parts. Even with the claims as a guide, however, the Examiner did not recreate the claimed invention.

8. Conclusion

The rejection of claims 1-20 should be reversed and the application promptly allowed.

Respectfully submitted,



Peter H. Priest
Reg. No. 30,210
Priest & Goldstein, PLLC
5015 Southpark Drive, Suite 230
Durham, NC 27713
(919) 806-1600

CLAIMS APPENDIX
(Claims Under Appeal)

1. A method of capturing operating software scheduling information during execution of operating software and recording said operating software scheduling information from within the operating software, the method comprising the steps of:

compiling operating software scheduling information capture software which is an integral part of the operating system, the operating software scheduling information capture software recording a history of operating software events as they occur, information related to the history being organized and stored as operating software program scheduling information relating to interactions between the operating system software and each of the programs and tasks managed by the operating system software, the scheduling information including indications of relative priorities of programs and tasks, indications of transfers of control from lower priority to higher priority tasks and identification of tasks waiting for execution at the occurrence of each operating software event;

invoking operating software scheduling information capture; and

recording operating software scheduling information while the operating software is executing, for relatively long duration storage in order to permit review of the scheduling information by a user, duration of storage being sufficient to allow data collected during an operating session to be retrieved and used after termination of the session.

2. The method as claimed in claim 1, further comprising:

scanning a specific bar code tag sequence with a retail store bar code scanner; and

selecting the information to be recorded by the operating software scheduling information capture software based upon the specific bar code tag sequence.

3. The method as claimed in claim 1, wherein the operating software scheduling information recorded includes information updated or maintained by the operating software in relation to the scheduling of a program.

4. The method as claimed in claim 1, wherein the operating software scheduling information recorded includes task identification, task priority, and task run-time length.

5. The method as claimed in claim 1, wherein the operating software scheduling information includes a task waiting count.

6. The method as claimed in claim 1, wherein the operating software scheduling information is recorded to a ledger.

7. The method as claimed in claim 6, wherein the ledger is at least one of a circular or fixed length ledger.

8. The method as claimed in claim 1, wherein the scheduling information includes at least one of the number of program schedules, program preempts, and interrupts.

9. The method as claimed in claim 1, wherein the scheduling information includes at least one of the highest priority attained, program identity and length of run-time.

10. The method as claimed in claim 1, wherein the scheduling information includes at least one of the lowest priority attained, program identity and length of run-time.

11. The method as claimed in claim 1, where the scheduling information includes at least one of the number of times in the idle loop and length of run-time.

12. The method as claimed in claim 1, wherein the scheduling information includes a sequential record of at least one of scheduled programs, priorities and events.

13. The method as claimed in claim 1, wherein the scheduling information includes at least one of the number and identity of programs waiting to run.

14. The method as claimed in claim 1, wherein the operating software scheduling information capture is invoked on an event occurrence.

15. A method of capturing operating software scheduling information during execution of said operating software and recording said operating software scheduling information from within the operating software, wherein said method is performed using operating software scheduling information compiled and integrated with the operating software, the method comprising the steps of:

invoking operating software scheduling information capture software, the operating software scheduling information capture software recording a history of operating software events as they occur, the history being organized and stored as operating software program scheduling information relating to interactions between the operating system software and each of the programs and tasks managed by the operating system software, the scheduling information including indications of relative priorities of programs and tasks, indications of transfers of control from lower priority to higher priority tasks and identification of tasks waiting for execution at the occurrence of each operating software event; and

recording operating software scheduling information while the operating software is executing for relatively long duration storage in order to permit review of the scheduling information by a user, duration of storage being sufficient to allow data collected during an operating session to be retrieved and used after termination of the session.

16. The method as claimed in claim 15, wherein said operating software scheduling information capture software is not resident on an external device.

17. The method as claimed in claim 15, wherein said operating software scheduling information capture software is not a separate task scheduled by an operating software scheduler.

18. A retail store bar code scanner employing operating software and operating software scheduling information capture software which operates during execution of said operating software to record a history of operating software events as they occur, the scanner comprising:

means for scanning a specific bar code tag sequence;

a processor for receiving and transmitting data; and

a memory coupled to the processor, the memory having stored therein sequences of instructions which, when executed by the processor, cause the processor to invoke operating software scheduling information capture software, the operating software scheduling information capture software being operative to record a history of operating software events as they occur, the history being organized and stored as operating software program scheduling information relating to interactions between the operating system software and each of the programs and tasks managed by the operating system software, the scheduling information including indications of relative priorities of programs and tasks, indications of transfers of control from lower priority to higher priority tasks and identification of tasks waiting for execution at the occurrence of each operating software event, and to record operating software scheduling information for relatively long duration storage in order to permit review of the scheduling information by a user, duration of storage being sufficient to allow data collected during an operating session to be retrieved and used after termination of the session wherein the specific bar code tag sequence selects the information to be recorded by the operating software scheduling information capture software.

19. The retail store bar code scanner as claimed in claim 18, wherein said operating software scheduling information capture software is internally processed on said processor.

20. The retail store bar code scanner as claimed in claim 18, wherein said operating software scheduling information capture software is not a separate task scheduled by an operating software scheduler.

EVIDENCE APPENDIX

None.

RELATED PROCEEDINGS APPENDIX

None.